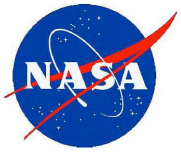

Biotic Prediction

Building the Computational Technology Infrastructure for Public Health and Environmental Forecasting

Software Requirements Trace Matrix

BP-SRTM-1.0

Task Agreement: GSFC-CT-1

November 30, 2003

1 Overview

1.1 Introduction

All requirements for the *Invasive Species Forecasting System* (ISFS), enumerated in the Software Requirements document (SRD), originated from many sources, are planned for delivery across multiple releases and will be validated through various techniques. A method is needed to annotate each requirement with information relating to these categories and track them through the development life cycle. The Requirements Trace Matrix serves this purpose. It provides the information to ensure that:

- the development plan will deliver product and services that satisfy all requirements
- each requirement is important & serves a purpose
- validation is thorough & appropriate
- deployment will yield a supportable & maintainable system

1.2 Document Versions

Date	Version	Description
Nov 30, 2003	1.0	Initial version relating to Milestone F

1.3 Referenced Documents

Document Title	Version	Date
Software Requirements (SRD)	1.6	2002-11-30
Test Plan (TP)	1.2	2002-11-30
Software Engineering / Development Plan	1.2	2002-09-26

1.4 Document Overview

The *Software Requirements Trace Matrix* documents and provides a tracking mechanism so requirements are properly, accurately & completely delivered in product & service. The SRD & SRTM are very tightly coupled, where updates to one may likely lead to an update to the other.

Section 1 provides an overview of requirements traceability. Section 2 maps each requirement found in the SRD across categories and disciplines, such as origin, dependency, release & testing method.

2 Requirements Traceability

This section contains the following per requirement:

- Traceability of the origin and dependency
- Verification methods that will be used to verify that a requirement has been adequately attained.
- The Build Plan which lays release drops for each requirement.

2.1 Source and Dependency

This criteria will be help to categorize, track & translate requirements into a work breakdown and user acceptance.

Source	reflects the origin of the requirement. This can be a user role, characteristic, organization, technical constraint, design goal or strategy.
Dependencies	denote what must exist in order for this requirement to be fulfilled & complete. These may be other requirements, tools & technologies and/or user knowledge.
Notes	will capture supplemental information not appropriate for either Source or Dependencies. Items noted here will be the fact that a requirement has been deleted or messages targeted at designers / implementers / testers important during these later stages of the development cycle.
Future Release	indicates that a requirement is to be analyzed, implemented & scheduled in a future, as yet to be defined release. It became apparent as an outcome of design & prototype exercises, that these system components need to further exploration & refinement. Each of these items is essential to the overall success of the project and will be targeted for a release in a future engineering plan.

2.2 Verification Methods

This criteria will be used as the basis for the *Software Test Plan*.

Verification Method	Description
Demonstration	The operation of the system, or a part of the system that relies on observable functional operation not requiring the use of instrumentation, special test equipment, or subsequent analysis.
Test	The operation of the system, or a part of the system, using instrumentation or other special test equipment to collect data for later analysis.
Analysis	The processing of accumulated data obtained from other qualification methods. Examples are reduction, interpolation, or extrapolation of test results.
Inspection	The visual examination of system components, documentation, etc.

2.3 Release Plan

Software builds & system releases will coincide with and include the functionality identified within existing milestones. Most requirements defined here are targeted for at least one release. Where a requirement is noted for more than one release denotes an enhancement to this requirement introduced in a previous release. The release-to-milestone relationships are:

Release 1 _ Milestone F
Release 2 _ Milestone G
Release 3 _ Milestone K

Certain requirements are not currently designated for any release, but are design goals (DG), which will provide additional functionality beyond key features (*"nice-to-haves"*) or enable system extensibility and wider audience participation.

Note: Milestones are defined in-detail w/in the Software Engineering Plan. The build plan in this requirements document (SRD) will be preserved only for the version pertaining to Milestone F. Further revisions of the plan will only be made in the Software Engineering Plan (SEP) document and not in the SRD. It would be inefficient to maintain two versions of the build plan and it logically belongs in the plan document.

2.4 Trace Matrix

ID	Requirement	Source	Dependencies	Release	Verification Method / Test Case	Notes
3.	**Functional Requirements**					
3.1	**User Interface**					
3.1.1	**Profile Database**					
3.1.1.1	**Error! Reference source not found.**	User-based logins & security. User-specific persistence.	Database to store user information. Method exists to maintain & make information available to users.	Release 2, 3	Demo	
3.1.2	**Roles**					
3.1.2.1	The system shall support various roles that control access to various capabilities of the system	User-based logins & security.	Database to store user information.	Release 2, 3	Demo	
3.1.3	**Graphical User Interface**					
3.1.3.1	The system shall include a Graphical User Interface (GUI) to support user interaction with the system.	User-base consists of various skill levels. Interface must be user-friendly.	User must have browser.	Release 2, 3	Demo	
3.1.3.2	The GUI shall dynamically construct personalized web pages based on the Profile Database and the User s Role.	User-base consists of various skill levels. Interface must be user-friendly.	Database to store user information.		Demo	Design Goal - Future Release
3.1.3.3	The GUI shall display predictive map and uncertainty map output.	System must provide meaningful & usable results, in various formats.	Algorithms, code & tools developed to produce such output.	Release 2, 3	Demo	
3.1.3.4	The GUI shall invite all system users to register in order to use the system based on their roles.	User-based logins & security.	Database to store user information. Method exists to maintain & make information available to users.	Release 2, 3	Demo	
3.1.3.5	The system shall allow the model builder to create new models w/in the Application functional layer.	System must be extensible & flexible. User-base consists of various skill levels.	Merged-data available and accessible.	Release 2, 3	Demo	
3.1.3.6	The system shall allow the model user to select from an assortment of modeling techniques and to modify model parameters.	System must be extensible & flexible.	Meta-data relating to modeling techniques & parameters resides in DB.	Release 2, 3	Demo	
3.1.4	**File Management**					
3.1.4.1	The user shall have the option of saving run results with annotations in personal repository.	User-specific persistence.	Database exists & accessible to store saved results.		Demo	Design Goal - Future Release
3.2	**Ingest**					
3.2.1	**Validation**					

ID	Requirement	Source	Dependencies	Release	Verification Method / Test Case	Notes
3.2.1.1	The system shall verify integrity, but is not required to validate data quality before ingest.	System must be extensible. User-base consists of various skill levels.	Method must exist to identify variables & data elements that must exist in & correlate with other data sets.	Release 1, 2, 3	Test / 3	
3.2.1.2	The system will accept data from an authoritative source.	Data integrity must preserved & enforced.	List of authoritative sources must be registered in the system		Demo, Analysis	Design Goal - Future Release
3.2.1.3	The system shall log all data sources.	System must be easily maintained & administered.	Logging levels must be defined. Method to persist logs must be agreed-on and developed.	Release 2, 3	Demo	Database logging preferable. Possibly model-related logging/output generated in a file with a pointer stored in db.
3.2.2	**Field Data**					
3.2.2.1	The system shall provide standard templates for ingesting field data in a tabular form.	System must be extensible & flexible. User-base consists of various skill levels.	Attributes of the ingestion process, categories of field data and the relationships between them must be ascertained.	Release 2, 3	Demo	Templates could be built dynamically based on metadata.
3.2.2.2	The templates shall include all required fields to be captured.	Interface must be user-friendly. Data integrity must preserved & enforced.		Release 2, 3	Demo	
3.2.2.3	The templates shall be in an accessible format.	Interface must be user-friendly.	Appropriate web-based presentation must be chosen.	Release 2, 3	Demo, Analysis	Relate templates to user-roles, certain may not be available based on access privileges.
3.2.3	**Remote Sensing Data**					
3.2.3.1	The system shall support ingest from external satellite data archives.	System must be extensible & flexible.	List of authoritative sources needed. Nature & method of storing data must be designed. Integrity checks, templates and logging must be in-place.	Release 2	Demo	Evaluate ECHO as the framework.
3.2.3.2	The system shall support ingest of user-supplied satellite data or airborne imagery from digital files.	System must be extensible & flexible.	Integrity checks, templates and logging must be in-place.	Release 2	Demo	Address user & system –level security of this type of ingested data.
3.2.3.3	The system shall support ingest of user-supplied data, also known as user-supplied layers.	System must be extensible & flexible.	Nature & method of storing data must be designed. Integrity checks, templates and logging must be in-place.	Release 1, 2, 3	Demo / 3	Address user & system –level security of this type of ingested data.
3.2.3.4	The system shall support ingest of user-supplied data files for ancillary layers.	System must be extensible & flexible.	Nature & method of storing data must be designed. Integrity checks, templates and logging must be in-place.	Release 2, 3	Demo	Address user & system –level security of this type of ingested data.
3.2.4	**Data Acquisition**					
3.2.4.1	The system shall support secured ftp-push for ingest of user supplied data.	System must be extensible & flexible. Interface must be user-friendly.	Determine firewall, network security constraints. Obtain & configure FTP server & storage device. Define appropriate file size thresholds	Release 2, 3	Demo	Address user & system –level access & storage limits.

ID	Requirement	Source	Dependencies	Release	Verification Method / Test Case	Notes
3.2.4.2	The system shall support automated secured ftp pull from external archives.	System must be extensible & flexible. User interface must be understandable, helpful and make task(s) easier.	Determine firewall, network security constraints. Obtain & configure FTP server & storage device. Define appropriate file size thresholds		Demo	Design Goal - Future Release
3.2.4.3	The system shall provide accounting and logging by requiring users name and password.	System must be easily maintained & administered.	Nature & method of storing data must be designed.	Release 2, 3	Demo	System interaction logged in DB structure, whereas model output may be in file-based logging due to its volume.
3.2.4.4	The system shall maintain a list of external archives and required data sets.	System must be easily maintained & administered. Interface must be user-friendly.	Identify archives and data sets. Determine how to persist & make available to the application.		Demo	Design Goal - Future Release
3.2.4.5	The interface between the system and each external archive will be thoroughly documented.	Interface must be user-friendly.			Ins	Design Goal - Future Release
3.2.5	**Monitoring and Reporting**					
3.2.5.1	The ingest subsystem shall monitor the number and volume of data brought into the system.	System must be easily maintained & administered.	Nature & method of storing information must be designed.	Release 2, 3	Test	Limits must be defined & enforced. Will violations result in rejection or notification.
3.2.5.2	The ingest subsystem shall produce data reports broken down by location, user, and external archive.	System must be easily maintained & administered.	Design method, frequency & design of reporting.	Release 3	Test	Reporting available to only certain roles.
3.2.5.3	The system shall generate and display associated metadata describing output files, runtime parameters, and performance statistics.	System must be easily maintained & administered.	Design method, frequency & design of reporting.	Release 3	Demo, Test	Reporting available to only certain roles.
3.3	**Pre-processing**					
3.3.1	**Merge Data**					
3.3.1.1	The system shall merge ingested datasets.	Interface must be user-friendly and helpful.			Test	Design Goal - Future Release
3.3.1.2	The system shall perform re-sampling if the input data are not at the same resolution.	Valid output must be ensured.	Determine criteria to program this behavior.		Test, Analysis	Design Goal - Future Release
3.3.1.3	The data shall be converted to a common analysis format.	Interface must be user-friendly and helpful.	Provide user ability to choose format. Define list of analysis formats.	Release 2, 3	Demo	
3.4	**Modeling**					
3.4.1.1	The system shall allow the ability to specify response and explanatory variables from the available databases.	System must be extensible & flexible. User interface must be understandable, helpful and make task(s) easier.	Field & Remote Sensed data ingestion must tag variables to be used in Model runs.	Release 2, 3	Demo	
3.4.1.2	The system shall provide graphical techniques to explore the relationships between these variables.	User interface must be understandable, helpful and make task(s) easier.	Relationships must be stored in DB. Graphic model to be chosen.	Release 2, 3	Demo	

ID	Requirement	Source	Dependencies	Release	Verification Method / Test Case	Notes
3.4.1.3	The system shall have the ability to "fit" models through Least Squares, or other optimization routines, such as Generalized Least Squares or Exhaustive Regression.	System must be extensible & flexible.	Code modules to perform these calculations must exist in the Backend layer. User interface provide option to perform routine.	Release 1, 2, 3	Demo, Test, Analysis / 1,2	
3.4.1.4	The system shall provide screening techniques to quantitatively assess which explanatory variables are related to the response variable.	User interface must be understandable, helpful and make task(s) easier.	Code modules to perform these calculations must exist in the Backend layer. User interface provide option to perform routine.	Release 2, 3	Demo, Test, Analysis	
3.4.1.5	The system shall calculate geospatial statistics (such as variograms).	System must provide meaningful & usable results.	Code modules to perform these calculations must exist in the Backend layer.	Release 1, 2, 3	Demo, Test, Analysis / 1	
3.4.1.6	The system shall incorporate spatial structure into the modeling.	System must be extensible & flexible.	Code modules to perform these calculations must exist in the Backend layer.		Demo, Test, Analysis	Design Goal - Future Release
3.4.1.7	The system shall be able to output to the user model results and relevant model diagnostics.	System must provide meaningful & usable results.	Secure & dynamic interface exists between Application & Backend Layer, so output can be "served-up" by Backend process(es).	Release 1, 2, 3	Demo / 1	
3.4.1.8	The system shall be able to create new models by processing ingested datasets through the modeling subsystem.	System must be extensible & flexible.		Release 2, 3	Demo	
3.4.1.9	The system shall allow the model user to select from an assortment of modeling techniques and to modify model parameters.	User interface must be understandable, helpful and make task(s) easier. System must provide meaningful & usable results.	Metadata for these functions exist in DB. User interface presents information appropriately.	Release 2, 3	Demo	
3.5	**Post-processing**					
3.5.1	**Re-projecting data**					
3.5.1.1	The system shall display an image of the output data.	System must provide meaningful & usable results, in various formats.	Decide/design program to generate image. What formats will be supported.	Release 1, 2, 3	Demo / 1	Where/how will image be persisted. Image size limits.
3.5.1.2	The system shall produce a data file suitable for re-projection by external COTS utilities.	System must provide meaningful & usable results, in various formats.	Agree-on what COTS utilities to support. Obtain these utilities to test with.	Release 1, 2, 3	Demo / 1	
3.5.2	**Data Overlay**					
3.5.2.1	The system shall overlay output data with other layers as requested.	Results able to be manipulated & formatted.	Can finite list of layers be generated? Obtain layers to integrate & test with.	Release 3	Demo	Does this layer data expire, invalid at some point in time.
3.5.3	**Metadata**					
3.5.3.1	Output data shall be packaged with appropriate metadata.	System must provide meaningful & usable results, in various formats.	Define & doc what metadata relates with Modeling techniques & parameters. Nature & method of	Release 1, 2, 3	Demo / 1	

ID	Requirement	Source	Dependencies	Release	Verification Method / Test Case	Notes
			storing information must be designed.			
3.5.3.2	Each output data set shall be assigned a unique identifier.	System must be easily maintained & administered. User-specific persistence.	Define methodology for generating unique id. Nature & method of storing information must be designed	Release 1, 2, 3	Demo / 1	
3.6	**Archive**					
3.6.1	**Database**					
3.6.1.1	The Archive shall have a database that will store pointers to the archived files.	User-specific persistence.	Calculate & obtain needed storage to support 12 months of files.	Release 2, 3	Demo	
3.6.1.2	The database shall be able to refer to internal (stored in the local File Store) and external files.	System must be extensible & flexible.	Method to store pointers to archives files must exist.	Release 2, 3	Demo	
3.6.1.3	All files (internal and external) shall be indexed with a unique file ID.	System must be extensible & flexible. User interface must be understandable, helpful and make task(s) easier.	Define methodology for generating unique id.	Release 2, 3	Demo	
3.6.2	**Internal File Store**					
3.6.2.1	Files shall be stored in a logically arranged directory structure.	User interface must be understandable, helpful and make task(s) easier.	Obtain storage to accommodate files. Decide method to logically order files	Release 2, 3	Ins	
3.6.3	**External Files**					
3.6.3.1	For external files, the archive system shall store a pointer that can be used to retrieve and stage the files for subsequent processing.	Sound architecture, development methods & appropriate technology will be employed.	Will retrieval method be chosen by user. Define how staged files will be stored in file system.	Release 2, 3		
4.	**Performance Requirements**					
4.1	**CT Project Scaling Milestones**					
4.1.1.1	Deliver canonical products 25X faster than the baseline implementation.	System will be an improvement over current methods.	Quantify baseline performance. Optimized modeling code for parallel processing, taking advantage of cluster array resource. Sufficiently powerful cluster array is available.	Release 1	Test, Analysis / 4	
4.1.1.2	Accommodate 10X more sample data at 25X the time required in the baseline implementation and 10X larger area at 2.5X the time required in the baseline implementation.	System will be an improvement over current methods.	Quantify baseline capacity. Disk storage available to warehouse sample data.	Release 2	Test, Analysis	
4.1.1.3	Deliver canonical products 200X faster than the baseline implementation on a 256-node cluster accommodating 100X more sample data 1000X faster than the baseline and 100X larger area 10X	System will be an improvement over current methods.	Obtain access to cluster.	Release 2	Test, Analysis	

ID	Requirement	Source	Dependencies	Release	Verification Method / Test Case	Notes
	the baseline and 100X larger area 10X faster than the baseline on a 1024-node cluster.					
4.2	**Security & Reliability**					
4.2.1	**User & sub-system security**					
4.2.1.1	The chosen transport protocol for the application subsystems to interface with the cluster must be secure and extensible.	Technical design constraint in accordance with NASA security policy	Chosen protocol can communicate to Cluster through firewall. May need to request firewall open a port.	Release 1	Test / 5	
4.2.1.2	The application will be multi-user safe	System must be extensible & flexible.			Demo, Test	Design Goal - Future Release
4.2.1.3	Evaluate whether a user account and user-specific stored data expire	Web-based application, supporting user-based logins & security.			Demo	Design Goal - Future Release
4.2.1.4	**Error! Reference source not found.**	User-based logins & security.	Database to store user information.	Release 2, 3	Demo	
4.2.1.5	**Error! Reference source not found.**	User-based logins & security. System must be extensible & flexible.	Database to store user information.	Release 2, 3	Demo	
4.2.2	**Resource Utilization**					
4.2.2.1	There needs to be a data clean-up scheme that specifies what data is to be discarded, what data is to be kept, and for how long.	User-specific data must be persisted. Data storage space limited.			Demo, Ins	Design Goal - Future Release
4.2.2.2	Internal storage requirements, including description of arrays, their size, their data capacity in all processing modes, and implied limitations of processing need to be determined & enforcement measures designed into the system.	User-specific data must be persisted. Data storage space limited.			Demo	Enhancement Opportunity
4.2.2.3	The system shall include utilities to monitor node status, utilization statistics, communication, etc. for troubleshooting and analyzing system performance.	System must be easily maintained & administered. Interface must be user-friendly.	Evaluate COTS to serve some functions		Demo	Enhancement Opportunity
4.2.2.4	The system shall monitor and manage CPU utilization by each user as needed.	System must be extensible & flexible.			Demo	Enhancement Opportunity

ID	Requirement	Source	Dependencies	Release	Verification Method / Test Case	Notes
4.2.3	**Stability & Maintenance**					
4.2.3.1	**Error! Reference source not found.**	System must be easily maintained & administered.	RMI protocol & method needs to be decided & designed		Demo, Test	Design Goal - Future Release
4.2.3.2	**Error! Reference source not found.**	System must be easily maintained & administered.	Will need science resource support		Demo, Test	Design Goal - Future Release
4.2.3.3	Pre-processing needs to be well defined and developed.	System must be easily maintained & administered.	Will need science resource support		Demo	Design Goal - Future Release
4.2.3.4	The need for a job controller to manage model runs on the cluster will be evaluated. Cluster management tools to work with the job controller and ISFS need to be well defined	System must be extensible & flexible.			Demo, Test	Enhancement Opportunity
4.2.3.5	The system shall provide the ability for model runs to be gracefully paused, resumed or shutdown in the middle of the run.	System must be extensible & flexible. Interface must be user-friendly and helpful.			Demo, Test	Design Goal - Future Release
4.2.3.6	The system shall support Linux. No designs to be portable to other platforms.	Technical team design decision	Available licenses & technical support	Release 1	Demo / 6	

Glossary

BP Biotic Prediction project (aka Invasive Species)
CT Computational Technologies project
GUI Graphical User Interface
ISFS Invasive Species Forecasting System
SEP Software Engineering / Development Plan
URL Uniform Resource Locator